SEUPD@CLEF: TEAM CLOSE
Temporal persistence of IR systems' performance

Gianluca Antolini
Marco Martinelli

Nicola Boscolo Cegion
Seyedreza Safavi

Mirco Cazzaro
Farzad Shami

Search Engines a.y. 2022/2023
Master Degree in Computer Engineering

TABLE OF CONTENTS

01 INTRODUCTION

PROBLEM

Performance of IR systems can deteriorate over time because web contents and user search preferences change.

TASK

Develop an efficient IR system and training it using *Qwant* training data (French & English):
user searches
web documents

GOAL

Test and measure the system's performance, ensuring that it doesn't decline with data coming from different/distant timestamps

02 COLLECTION

- Data Source: *Qwant* (search engine) search logs
- Queries: Filtered for spam and low document count
- Documents: Relevant documents extracted from SERPs, non-relevant documents randomly sampled from *Qwant* index
- Relevance Estimates: Obtained through user implicit feedback using a click model

COLLECTION

- Training Data: 672 queries with 9,656 assessments
- Heldout Data: 98 queries with 1,420 assessments
- Document Corpus: 1,570,734 web pages
- Relevance Distribution: 73% non-relevant, 21% relevant, 6% highly relevant
- Test Collections:
  - Short-term Persistence Sub-task: 1,593,376 documents, 882 queries
  - Long-term Persistence Sub-task: 1,081,334 documents, 923 queries

03 OUR SYSTEM

Parser

Pre-processing the documents by cleaning them and removing noise

Analyzer

Analyzing parsed documents using techniques such as tokenization, stemming, ...

Indexer

Indexing documents keeping necessary fields (id, body, preparing them for the search phase

Searcher

Searching through indexed documents, retrieving and ranking them based on relevance

PARSER

Custom ClefParser class implemented to remove various types of noises present in the documents.

Trial and error workflow/methodology (considering improvements in **MAP**):

1. Examination of large statistical sample size of collection documents in the collection to decide types of noises to be removed.
2. Implementation and run of the parser.
3. Results stored and sample of parsed documents analyzed to restart the procedure.

Final Parsed Document structure:

- *id* : document identifier
- *body* : parsed content of document

PARSER

Types of noises identified and **removed**:

- ***JavaScript* scripts**
- ***HTTP* and *HTTPS* URIs**
- *HTML* tags and *CSS* stylesheets
- *XML* and *JSON* codes
- Meta tags and document properties
- Navigation menus
- Advertisements
- Footers
- Social media handlers
- Hashtags and mentions
- **Word patterns** (e.g. *word1_word2* or *word11.word2* or *word1:word2*)

GSON

Integration of Gson Library to parse a JSON file containing documents.

Advantages:

- Efficient parsing of JSON files.
- Seamless conversion of JSON data into Java objects.
- Effortless manipulation and integration of the query expansions into the application.

This ensured an overall increase in the system’s performance

Gson

ANALYZER

Fully customizable class to analyze documents using different approaches.

Based on experiment results with different parameters and by trying both English and French dataset, the following are the parameters used to get the best results:

- French dataset
- FrenchLightStemFilter
- StandardTokenizer
- LowerCaseFilter

- Minimum token length: 2
- Maximum token length: 15
- French word stoplist (662 French words): built upon popular French stoplist and most frequent stopwords in the collection.

INDEXER

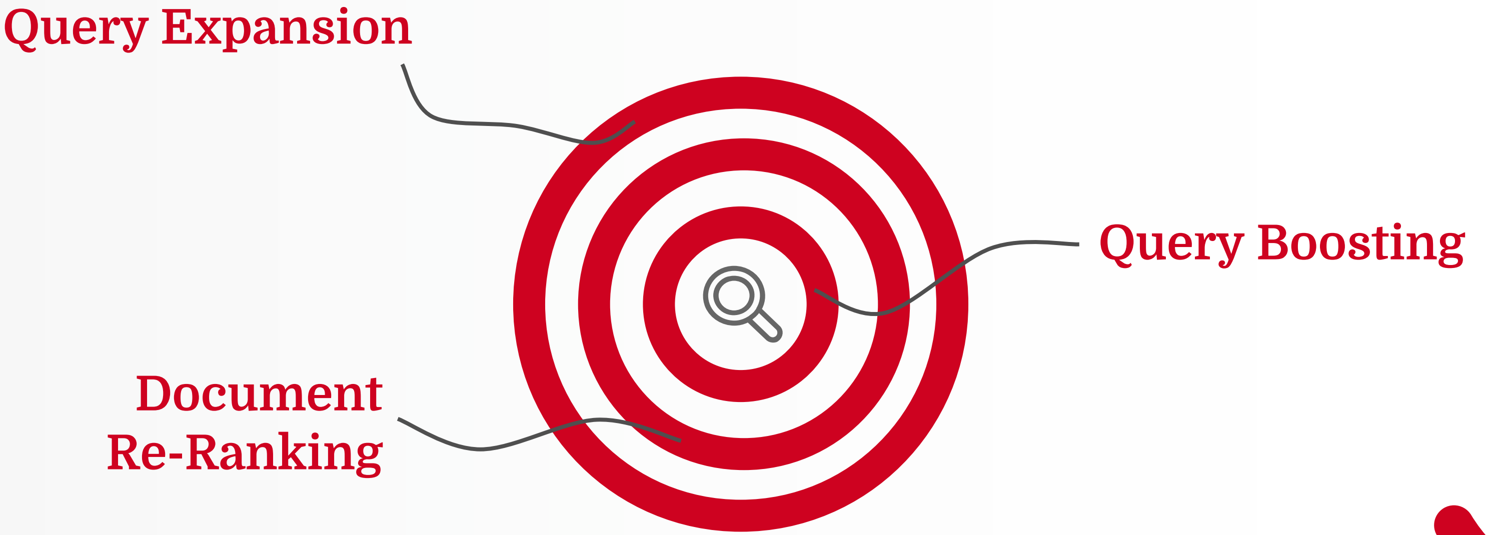
Invokes Parser and Analyzer, processes the documents with both, and then indexes them as a ParsedTextDocument objects (defined in the Parser).

It takes the following inputs:

- Analyzer
- Similarity
- RamBufferSizeMB
- IndexPath
- DocsPath
- Extension
- CharsetName
- ExpectedDocs
- DpCls

SEARCHER

Retrieve relevant information by analyzing user queries and searching through indexed documents, returning a ranked list of matching documents.



QUERY EXPANSION

During the search function, Query Expansion performed by generating new queries from the original ones.

Python script that utilizes OpenAI's Text Completion Endpoints to generate expanded terms for each query.
Expansions stored in a .json file called "result."
We used the DaVinci model with a temperature parameter of 0.6 for optimal results.

OpenAI

QUERY BOOSTING

To assign higher relevance to specific query terms or queries.

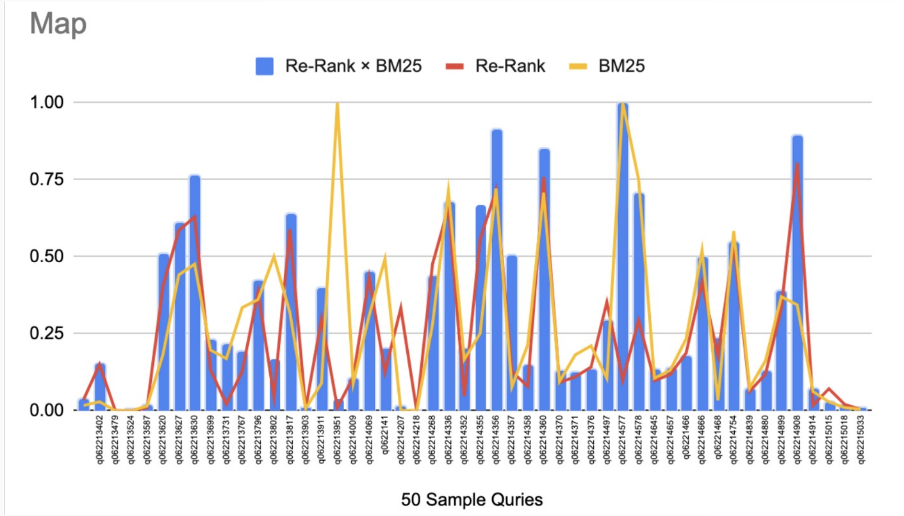
Our approach consists of constructing BooleanQueries in the search function:

- Each query has query expansions added with SHOULD clause.
- A main query with the MUST clause.
- Main query boosted using Lucene's BoostQuery with tuned boost value * number of expansions.
- Boost value fine-tuned to 14.68 through a trial and error for parameter optimization.

DOCUMENT RE-RANKING

- Rank documents retrieved by the Searcher using all-MiniLM-L6-v2, a 384-dimensional Sentence Transformer model.
- Initialize Re-Ranker in the Searcher's constructor and create a predictor for inference.
- During search function, embeddings created for documents using predictor, similarity calculated between query and documents.
- Scores multiplied by document's BM25Similarity and cosine similarity.
- Tested parameter combinations and determined that multiplying document's score by BM25Similarity with cosine similarity yields the best results.

DOCUMENT RE-RANKING



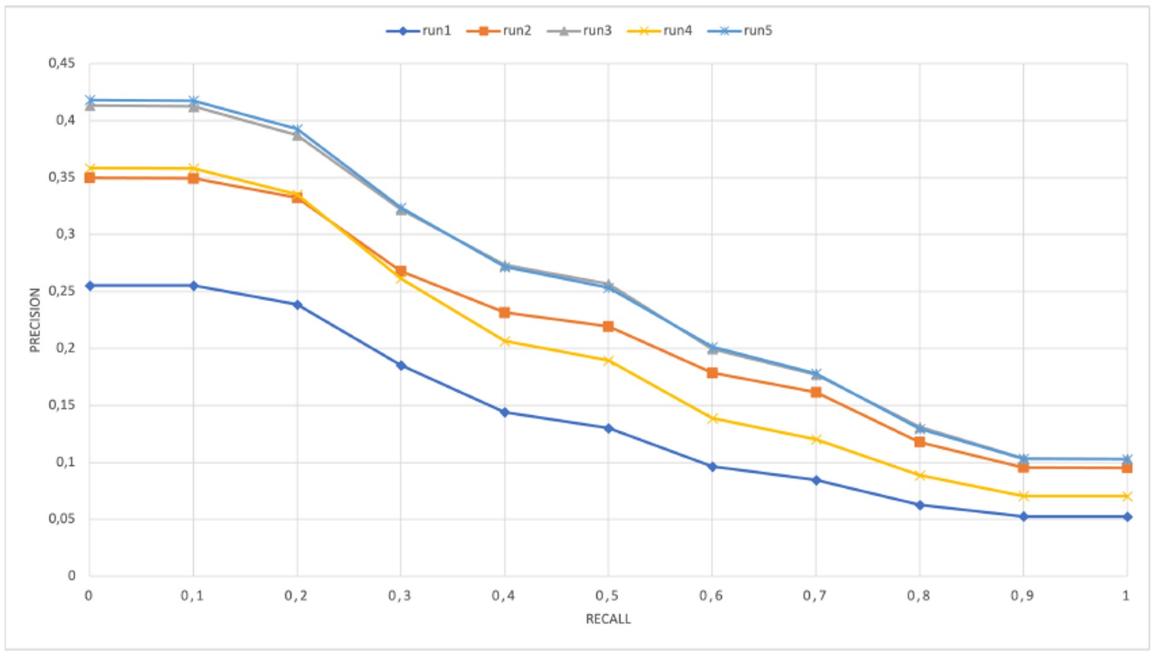
Re-Ranking performed on a sample of 50 queries (MAP scores).

04 RESULTS

The following slides provide the results of our **best** performing runs, with both French and English dataset

Parameter	**Run 1**	**Run 2**	**Run 3**	**Run 4**	**Run 5**
Token Filter	Porter-StemFilter	FrenchLight-StemFilter	FrenchLight-StemFilter	PorterStem-Filter	FrenchLight-StemFilter
Tokenizer	Standard	Standard	Standard	Standard	Standard
Length Filter	2-15	2-15	2-15	2-15	2-15
Stop Filter	"*long-stoplist-fr.txt*"	"*long-stoplist-fr.txt*"	"*long-stoplist-fr.txt*"	"*long-stoplist.txt*"	"*new-long-stoplist-fr.txt*"
Lower Case Filter	Yes	Yes	Yes	Yes	Yes
Similarity	BM25	BM25	BM25	BM25	BM25
Query Expansion	No	Yes	Yes	Yes	Yes
Re-ranking	No	No	Yes	Yes	Yes

RESULTS



run1
run2
run3
run4
run5
PRECISION
0,45
0,4
0,35
0,3
0,25
0,2
0,15
0,1
0,05
0
0
0,1
0,2
0,3
0,4
0,5
0,6
0,7
0,8
0,9
1
RECALL

RESULTS

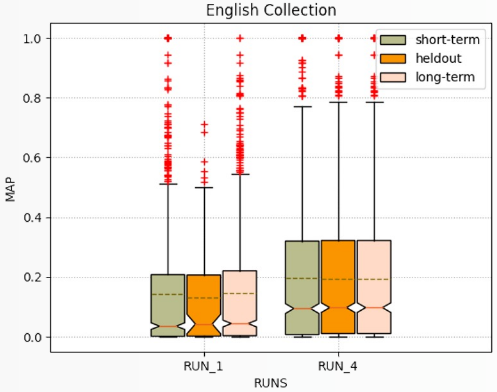
metrics	run1	run2	run3	run4	run5
num_q	657	669	669	667	667
num_ret	646525	658446	658347	652222	657903
num_rel	2550	2611	2611	2603	2600
num_rel_ret	1772	2182	2191	1866	2232
map	0.1307	0.2022	0.2335	0.1856	0.2351
gm_map	0.0117	0.046	0.061	0.0239	0.0629
Rprec	0.1041	0.1697	0.1989	0.1654	0.2022
bpref	0.3142	0.3734	0.3869	0.3466	0.3861
recip_rank	0.2436	0.3287	0.3891	0.3441	0.3945
iprec_at_recall_0.00	0.2553	0.3499	0.4134	0.3584	0.4182
iprec_at_recall_0.20	0.2387	0.3324	0.3873	0.3353	0.3927
iprec_at_recall_0.40	0.1441	0.2316	0.2732	0.2066	0.2716
iprec_at_recall_0.60	0.0965	0.1786	0.1996	0.1388	0.2014
iprec_at_recall_0.80	0.0628	0.1178	0.1311	0.0887	0.1295
iprec_at_recall_1.00	0.0525	0.0954	0.1031	0.0704	0.1028
P_10	0.0848	0.1296	0.1435	0.1126	0.1432
P_100	0.0186	0.0256	0.0268	0.0222	0.0268
P_1000	0.0027	0.0033	0.0033	0.0028	0.0033
recall_10	0.2166	0.3352	0.367	0.2849	0.3621
recall_100	0.4718	0.6426	0.6714	0.5536	0.6723
recall_1000	0.6816	0.8192	0.8218	0.7004	0.8392
infAP	0.1307	0.2022	0.2335	0.1856	0.2351
gm_bpref	0.0152	0.0387	0.0405	0.022	0.038
utility	-978.6621	-977.701	-977.5262	-972.2489	-979.6687
ndcg	0.2719	0.3655	0.3924	0.3291	0.3982
ndcg_rel	0.236	0.3119	0.3416	0.2939	0.3471
Rndcg	0.1708	0.2387	0.2657	0.2271	0.2714
ndcg_cut_5	0.1285	0.1908	0.2232	0.1854	0.2269
ndcg_cut_10	0.1609	0.2426	0.2739	0.2227	0.2758
ndcg_cut_100	0.2351	0.3349	0.3652	0.3016	0.3678
ndcg_cut_1000	0.2719	0.3655	0.3924	0.3291	0.3982
map_cut_10	0.1046	0.1665	0.1975	0.1556	0.1993
map_cut_100	0.1284	0.2	0.2315	0.1836	0.2328
map_cut_1000	0.1307	0.2022	0.2335	0.1856	0.2351

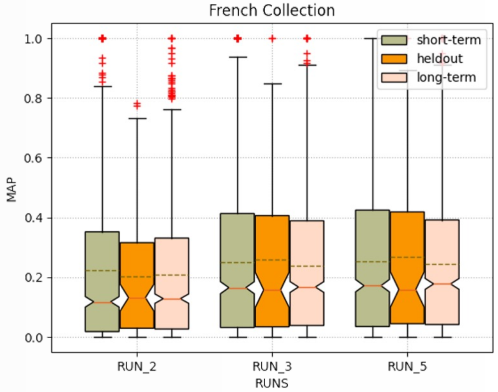
RESULTS

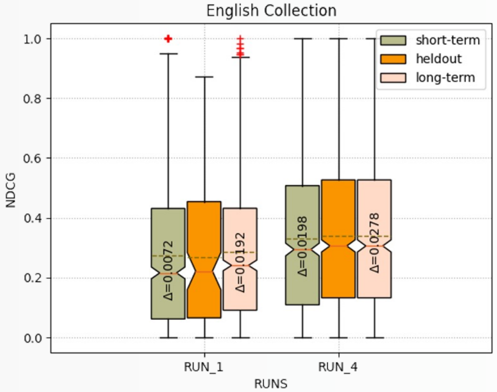
heldout						
run	language	type	map	p@10	NDCG	recall
run2	FR	QUEREXPANSION	0.2029	0.1367	0.3725	0.8312
run3	FR	RERANKING	0.2595	0.1541	0.4166	0.8348
run5	FR	SBERT_BM25	0.2675	0.1561	0.4318	0.8726
run1	EN	JSCLEANER_BM25	0.1299	0.0897	0.2674	0.6381
run4	EN	RERANKING_ENGLISH	0.1822	0.1122	0.3113	0.6279

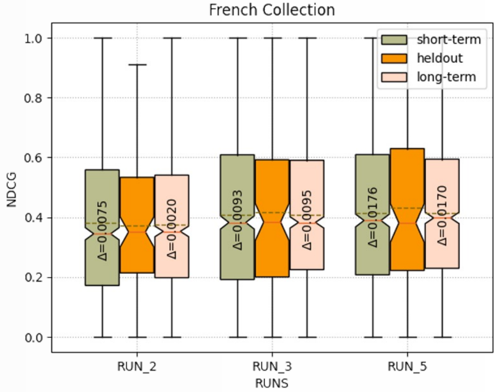
Short term						
run	language	type	map	p@10	NDCG	recall
run2	FR	QUEREXPANSION	0.2215	0.1326	0.3800	0.8164
run3	FR	RERANKING	0.2511	0.2171	0.4073	0.8142
run5	FR	SBERT_BM25	0.2540	0.1497	0.4142	0.8360
run1	EN	JSCLEANER_BM25	0.1438	0.0902	0.2746	0.6566
run4	EN	RERANKING_ENGLISH	0.1956	0.1145	0.3311	0.6804
Long term						
run	language	type	map	p@10	NDCG	recall
run2	FR	QUEREXPANSION	0.2067	0.1423	0.3745	0.8312
run3	FR	RERANKING	0.2388	0.1555	0.4071	0.8336
run5	FR	SBERT_BM25	0.2437	0.1594	0.4148	0.8540
run1	EN	JSCLEANER_BM25	0.1450	0.0975	0.2866	0.6906
run4	EN	RERANKING_ENGLISH	0.1930	0.1258	0.3391	0.7119

Statistical Analysis

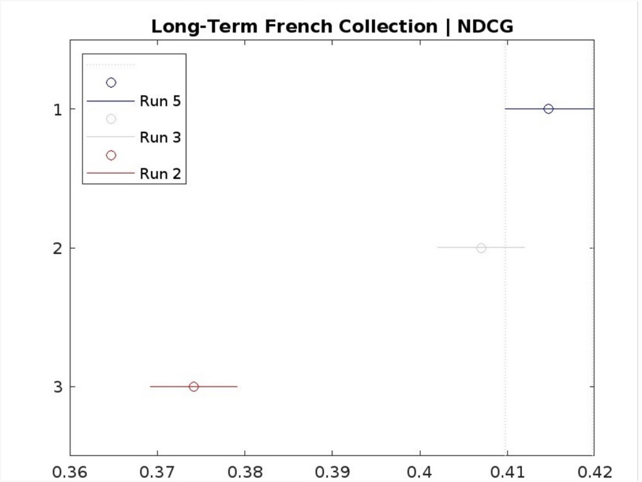


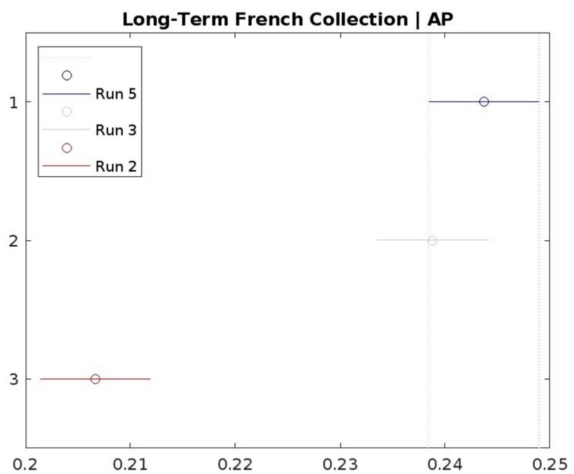






Statistical Analysis | FR

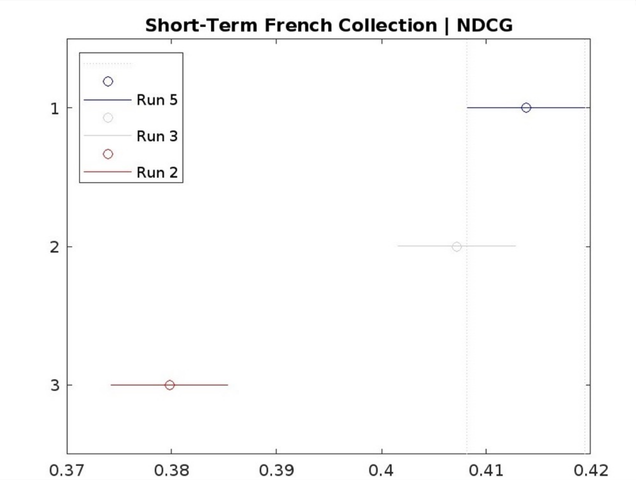


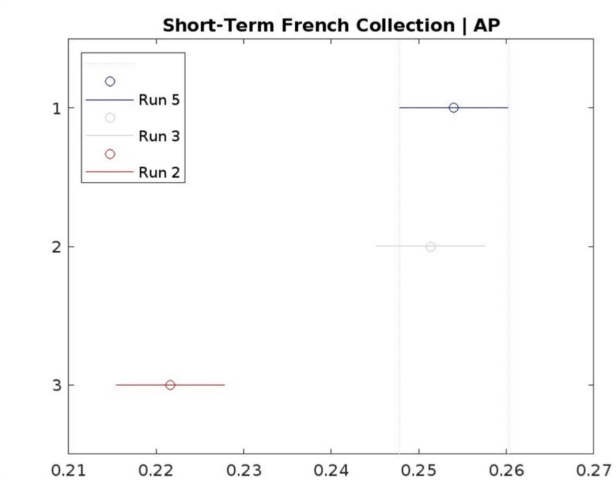


Group A	Group B	Lower Limit	A-B	Upper Limit	P-value
5	3	-0.0023	0.0077	0.0177	0.1661
5	2	0.0305	0.0405	0.0505	1.16E-21
3	2	0.0228	0.0328	0.0428	3.84E-14

Group A	Group B	Lower Limit	A-B	Upper Limit	P-value
5	3	-0.0057	0.0049	0.0154	0.523
5	2	0.0265	0.037	0.0476	0
3	2	0.0216	0.0322	0.0427	0

Statistical Analysis | FR

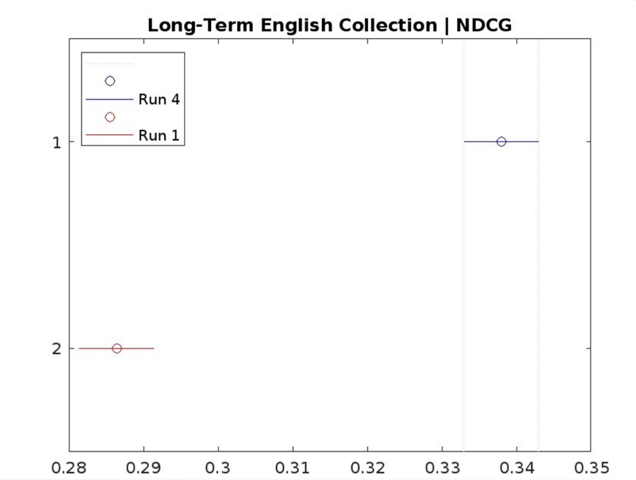


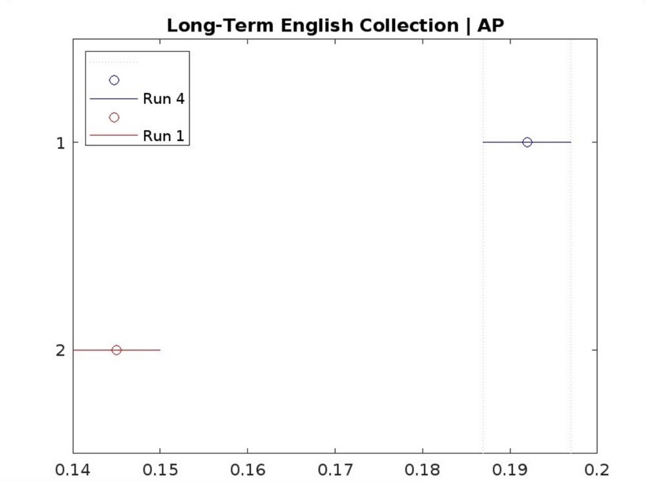


Group A	Group B	Lower Limit	A-B	Upper Limit	P-value
5	3	-0.0047	0.0066	0.0178	0.355
5	2	0.0227	0.034	0.0452	3.86E-12
3	2	0.0162	0.0274	0.0386	3.33E-08

Group A	Group B	Lower Limit	A-B	Upper Limit	P-value
5	3	-0.0098	0.0027	0.0151	0.8701
5	2	0.0199	0.0324	0.0448	3.15×10^{-9}
3	2	0.0173	0.0297	0.0422	6.48×10^{-8}

Statistical Analysis | EN

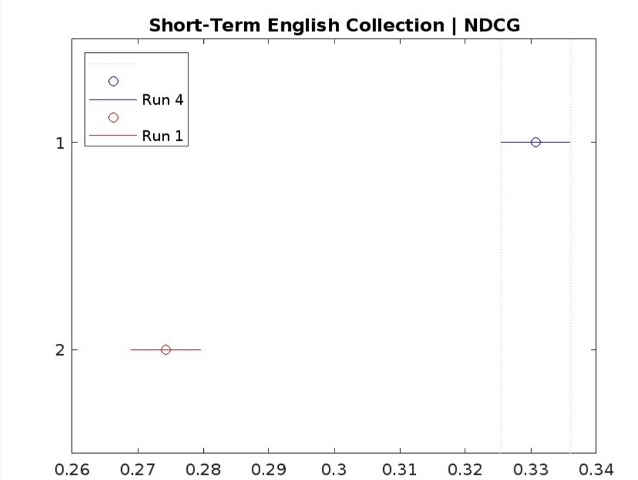


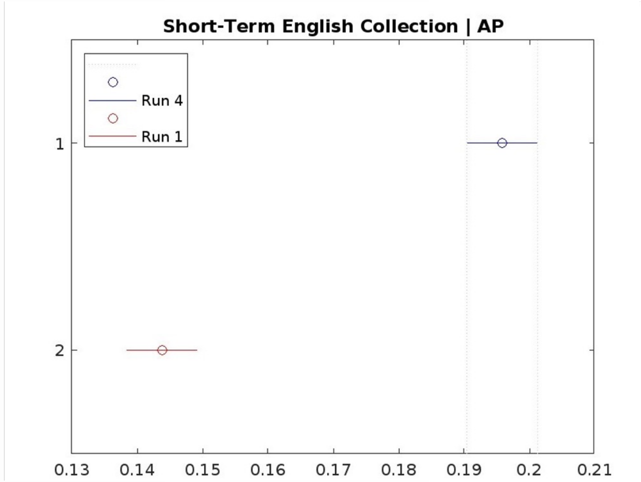


Group A	Group B	Lower Limit	A-B Difference	Upper Limit	P-value
4	1	0.0415	0.0515	0.0615	6.77×10^{-25}

Group A	Group B	Lower Limit	A-B Difference	Upper Limit	P-value
4	1	0.0369	0.0469	0.057	2.13×10^{-20}

Statistical Analysis | EN





Group A	Group B	Lower Limit	A-B Difference	Upper Limit	P-value
4	1	0.0457	0.0564	0.0671	0.00×10^{0}

Group A	Group B	Lower Limit	A-B Difference	Upper Limit	P-value
4	1	0.0412	0.052	0.0628	1.05×10^{-21}

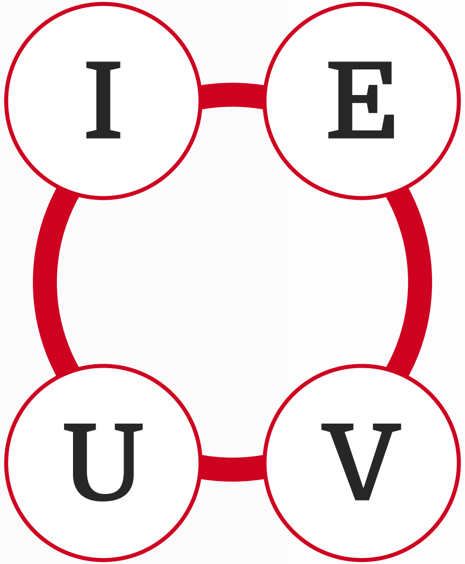
05 CONCLUSIONS
& FUTURE WORK

Improving Re-ranking

Diversify scores, explore BERT models, fine-tune SBERT for relevance and document retrieval enhancement.

Enhancing Query Expansion

Better ChatGPT prompts and explore alternative Large Language Models techniques.



Utilizing Document Links

Enhance search results with link details and domain authority from URL keywords.

Vector-based Document Indexing

Investigate indexing documents as vectors for faster re-ranking, sacrificing one score in the process.

THANKS!

Do you have any questions?